Chess.com Rated Match-making Analysis Report

November 2022

INTRODUCTION

Let's dive into the world of an indoor sport. The popularity of which has leaped several fold in the past couple of years. The title is Chess. The player volume on online platforms has been spiking up rapidly providing data analysts the resource & opportunity to delve deep into the game for getting insights. More and more fun variant modes are being created. Recently, there has also been a controversy in a World Chess Championship tournament which has led statisticians from many places to dig into the databases of concerned player's game history. Despite that it has helped the game gain even more popularity. There are many new platforms and apps being developed too. Two of the most popular ones are Chess.com and Lichess.org.

OBJECTIVE

To test the perfomance of the current live player vs player rated match-making algorithm of Chess.com, the outcome and volume of games played on Chess.com shall be analysed to be able to better pass a judgement regarding the quality of the algorithm and whether or not it needs significant optimization. The observations must consist of elo of both players and the result of the game along with time settings of the games played. The analysis will be narrowed down to one of the most widely played time setting but giving more priority to skill-based games and thus not considering very short span of games. This would include data wrangling and tinkering. Next, the result form needs to be deciphered from several reasons (checkmated, stalemated, insufficient material, insufficient time, resigns, repitition, 50moves, abandon), etc. Result will be categorized in 3 states: 'White-win', 'Black-win', 'Draw and respectively ennumerated into +1, -1, 0. The absence of this categorical data will be treated as missing features in the data set that will all be handled. The elo rating matched between 2 players will be used to derive difference in ratings and mean of ratings. These will be compared and plotted to get insight on where the winners, losers and draw-doers lie. Furthermore, we will get information on what match rating maximum games out of the given data are being played on, where we can conduct reliable testing of our hypothesis of whether or not we can determine that a player with pieces (white or black) is better.

DATA

We are going to look at a chess.com games data set uploaded on Kaggle by a user named 'ADITYAJHA1504'. It was last updated a year ago as of Nov '22. The direct link to it is here - https://www.kaggle.com/datasets/adityajha1504/chesscom-user-games-60000-games. File name is club_games_data.csv, size 166.67MB.

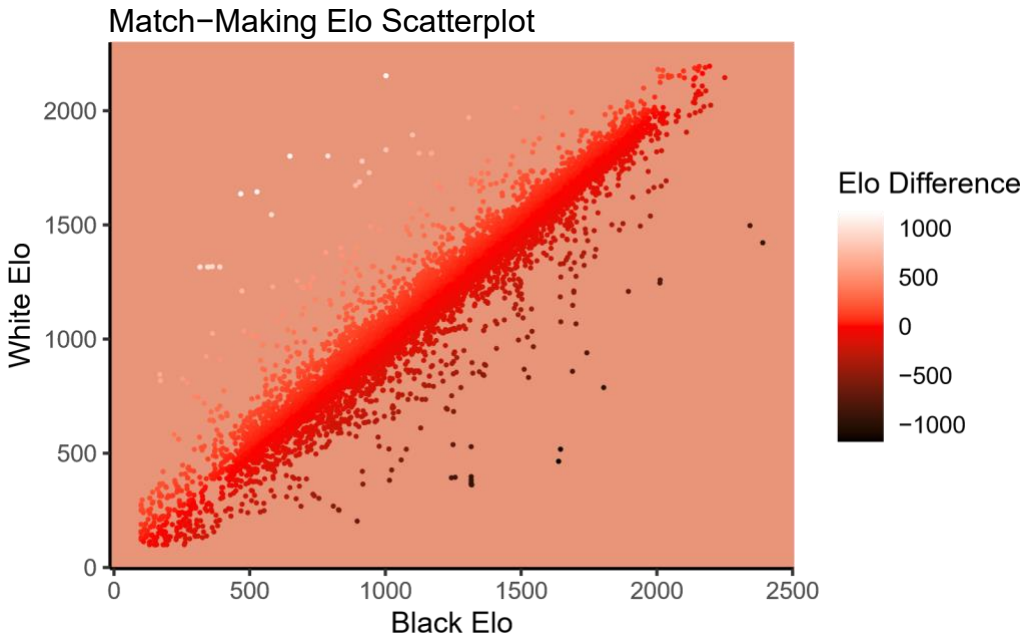
It has 60,000+ games and 14 columns (all string) out of which we would require white_rating, black_rating (skill levels), white_result (automatically tells black_result), time_class (categories of different time_controls), time_control (what time setting is the match being played on), rules (game variants), rated (casual/competitive). At first, we will examine the unique values in categorical features to determine how to filter the data. We have 5 different game variants being played based on rules column - chess, chess960, threecheck, crazyhouse and kingofthehill. We pick the standard chess games. We have 13 different game outcomes "win", "checkmated", "timeout", "resigned", "abandoned", "timevsinsufficient", "repetition", "insufficient", "stalemate", "agreed", "threecheck", "kingofthehil", "50move". We narrow them down according to game variant outliers and deterministic results. These white's results are then simplified and mapped to 2 new features initially treated as missing shown as follows- 'white-win' = +1 = win, 'black-win' = -1 = checkmated, timeout, resigned and 'draw' = 0 = timevsinsufficient, repetition, insufficient, agreed, stalemate. We also eliminate time class column after getting an overview of its unique values namely - daily, rapid, bullet, blitz. Time controls has 62 unique game time settings so we tabluated and sorted it in decreasing order of frequency to find the most played rapid game time setting in seconds.

Based on that, one can tell that an overwhelming proportion of games are being played at the top 2 time control settings of 60 seconds and 600 seconds. Since a 60 second game falls into the bullet category of time_class, we have to discard it due to less skillful influence of elo on the outcome. We pick the 12,773 games for 600 seconds i..e. 10 minute time control from rapid time_class category and will analyze the same. We discard all other.

ANALYSIS & RESULTS

We plot a scatter plot between elo of player with black pieces (black_elo) on x-axis vs elo of player with white pieces (white_elo) on y-axis and we assign difference in their elo (delta_elo = white_elo - black_elo) variable as the coloring parameter for the points plotted. This can be negative. The minimum and maximum value it can take when players are matched in standard rated game is -1000 and +1000 respectively. Naturally, a +1000 means player with white pieces leads with a 1000 elo points so he is more or less destined to win, therefore we colour

this extreme as white. For similar reasons, we colour the other extreme as black. And at the midpoint of delta_elo = 0 when white_rating and black_rating are both equal, we expect highest likeliness of draw which we colour by red. The plot background is coral so that the transition of white - red - black is visible.

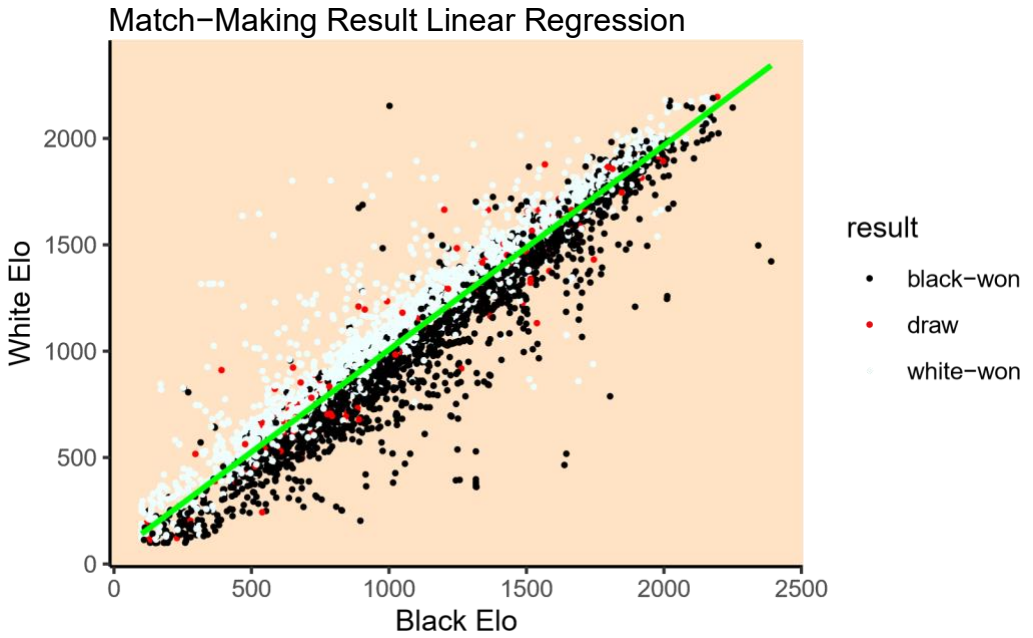


We can see that in many cases, Chess.com has matched the players nearly equally on roughly y=x line. Secondly, it can be noticed that there are fewer points at the top-right end than the bottom-left end and most points accumulate in the centre. It's because more players are average in skill and all new players start from 1500 elo while few people can play as bad as dropping their elo towards 100 - the minimum possible value and lastly high elo players are least in number which makes them pro at the game. Next up, one can notice that the points are mostly red and its dark and light shades. It's because majority of the matches occured approximately near the y=x line which has maximum likeliness for the match to end in a tie but will we really see so many ties? Apart from this, there are several outliers though and they have crystal clear bias. So one of the players are extremely skilled as compared to the other and thus those matches could be one-sided.

Let's test this by plotting a Linear Regression to get a mean matchmaking line but this time we will replace the coloring variable of delta_elo by actual result of the game. We follow the coloring rule synonymous to previous. There are 3 possible values in actual outcome: whitewin, black-win and draw. These will be white color, black color and red color

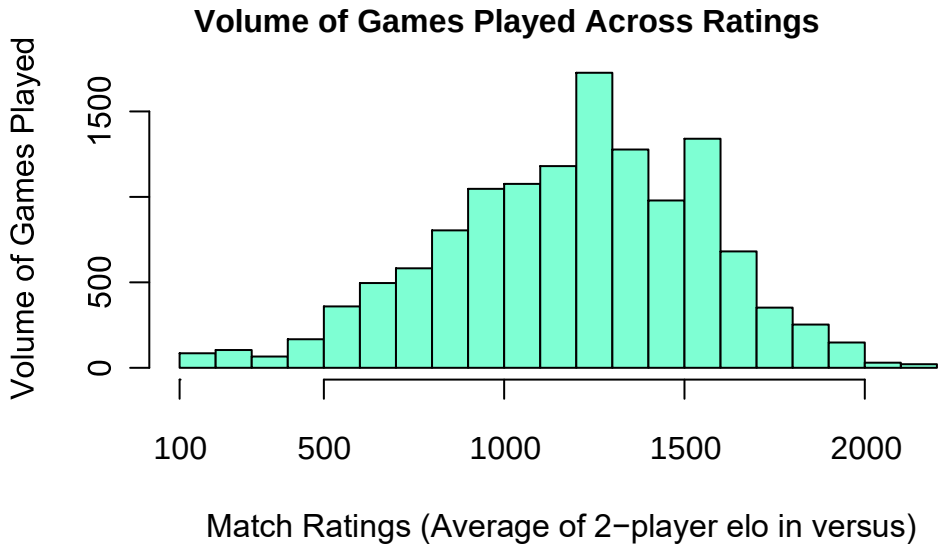
respectively. Now we will see how much quantity of the red scatterplot (expected draws) become white and black in color.

```
`geom_smooth()` using formula 'y ~ x'
```

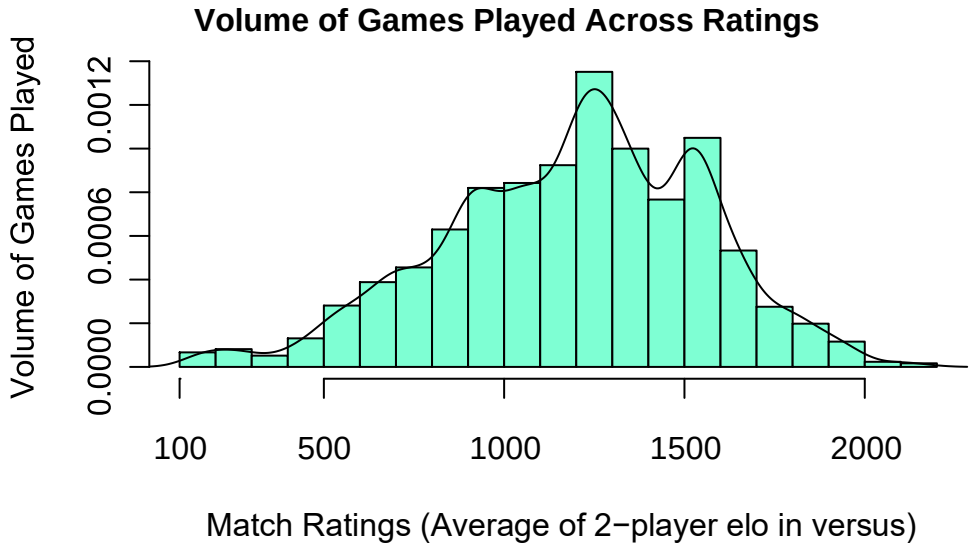


Not nearly as many draws as one might have expected from the scatterplot. And a very obvious observation that there are mostly white dots above the linear regression line and mostly black dots below the line, simple because higher elo wins over lower elo more number of times. Interestingly there are few outliers not following their probabilistic color meaning few of them won despite the odds much against them. But timeouts and blunders in chess could be the most contributing reason to them. It's fair to see the red dots averaging about the linear regression line. Most interesting fact that can be noticed from this linear regression graph is that the density of black dots penetrating above the green line increases as we climb on higher elo (roughly after 1200 elo on x-axis).

Let's visualize through a histogram - how many matches occur at all these different rating ranges to get a better idea of the above 2 plots' distributions backed by quantity of matches played. We will have x-axis as the match_rating column (i..e. the mean elo of the game) obtained from data wrangling.



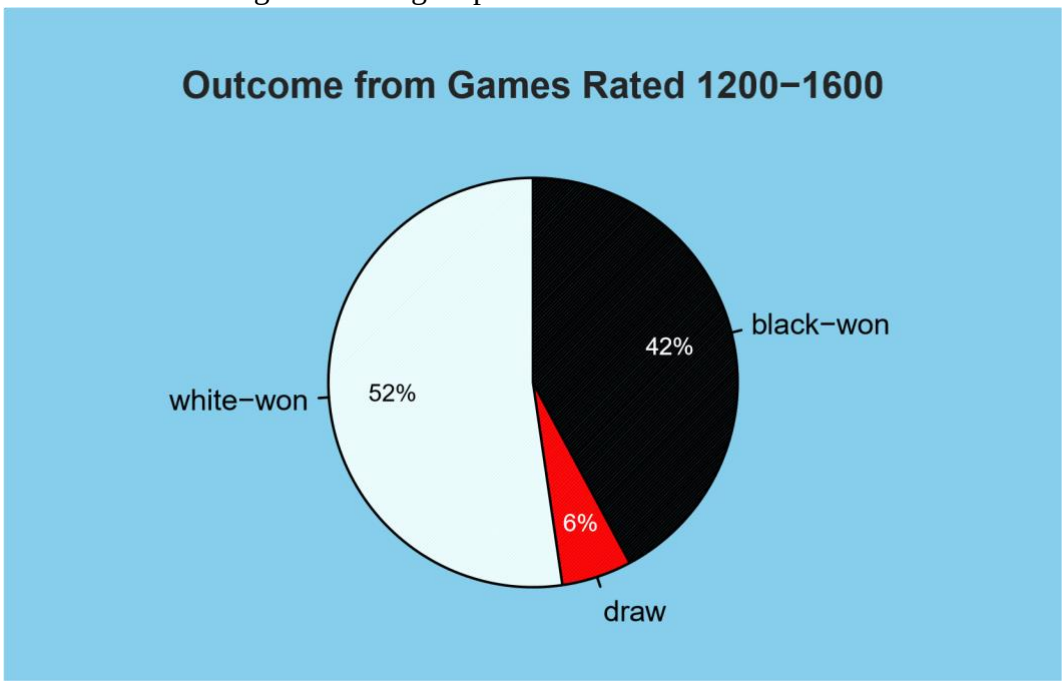
The number of matches being played see an increase and then a decrease as we move across the ratings in either direction. The maximum volume of matches seems to be occuring between a rating range of 1200 to 1600 where we could test our hypothesis for it to be most reliable at the peak of competition. Interestingly a sudden spike after 1500 before the volume goes down. Now Let's have a look at the Kernel Density Estimate curve of this histogram to get an idea of the highest growth rates of frequency.



The maximum growth rate of the curve is in elo ranges of 800-900, 1100-1200 and 1400-1500 as the curve is looking most up-steep there. We will restrict our analysis from here on, to the

elo range of 1200-1600. We also need the elo difference to be small to do fair comparison of white vs black pieces winning probabilities. We will settle on a max elo difference of 10.

After we do the necessary filtering of elo range (1200 to 1600) and elo difference (-10 to 10), we find 1065 games following these criteria. It will be sufficient to conduct a hypothesis test as observations > 1000. But before the test, let's have a look at the proportion of both wins and draws on these games through a pie chart.



As one can clearly tell that the winning proportions are distinct for white and black pieces with white having more. The great thing is that there are only 6% draws in these games so it will be worth testing whether or not one of the pieces has an advantage over the other due to Chess.com's matchmaking algorithm.

Let us discuss about a null hypothesis first. The elo rating given by Chess.com changes with each game unless it's a draw. And since there are only 6% draws out of the 1065 games being analyzed, we are left with 94% of decisive outcome matches. If a player wins, their elo will rise and if they lose, it will fall. Moreover, since we are in the 1200-1600 elo range where the rated games being played are highest, that means players likely have played many games in this range and so the ones whose elo is higher than the other could be of truly higher skill. To make it even more competitive, we have restricted the elo difference in players matched to a maximum of 10 which means the delta_elo can range from -10 to +10. Thus, if the elo difference is positive i..e. in favour of white piece, then that match could have a likely outcome

of white piece player as winner if the elo given by Chess.com to players is any significant combined with its matchmaking algorithm to pair players for rated games based on their elos.

Null Hypothesis: “Actual outcome of a game can be predicted by (+-) delta_elo/match_rating. If this value is positive, white has more likelihood of winning. If the value is negative, black has more likelihood of winning.”

Alternate Hypothesis: “Actual outcome of the game cannot be predicted by difference in elo ratings. Chess.com has not reached high precision to assign accurate elo to players.”

(+-) delta_elo is signed difference in elo of 2 players matched together: (white_elo - black_elo) & match_rating is mean of both players’ elos. We divided by match_rating so that we get values between -1 to 1 which are comparable to the quantified results column with values = {1,0,1} perceived as black-win, draw, white-win respectively. Let us now compare the expected outcome with the actual outcome to test the null hypothesis.

```
    Welch Two Sample t-test

data: expectedOutcome and actualOutcome t = -3.4225,
df = 1064, p-value = 0.000644 alternative
hypothesis: true difference in means is not equal to
0 95 percent confidence interval:
 -0.15956723 -0.04327435
sample estimates:
   mean of x   mean of y
-0.0000123359 0.1014084507
```

Since p.value observed is 0.000644 which is very less than 0.05, null hypothesis can be successfully rejected! Alternate hypothesis is supported by analysis of 1065 games, that outcome of a game CANNOT be predicted based on difference in elo.

LIMITATIONS

The rating range analyzed was 1200-1600 due to highest number of matches being played in it. Other ranges could affect the test results with possibility of extreme outcomes. And since at different rating ranges, players are of different skill level, the precision of test would be highest while analyzing matches of highest rated players as they do limited blunders in the game. There is scope of continuation of research. Each rating range can be analyzed in subgroups to further conclude any possible advantage white can have in higher rated games using agglomerative clustering (bottom-up) approach.

CONCLUSION

Alternative hypothesis states that true difference in means is not equal to 0 with 95% confidence interval as (-0.159, -0.043) suggesting white has a slight advantage. Reason: true mean difference = true mean(expected outcomes) - true mean(actual outcomes). If actual outcome value is +1 (white won) then the true mean difference can more likely be a negative. Thus white has a slight advantage and which is also proved in the pie chart where white wins 52%, black wins 42% and rest are draw. Therefore the elo provided by Chess.com is less significant than assumed and it's matchmaking algorithm certainly has scope of optimization to improve upon.

APPENDIX CODE

```
#0. PRELIMINARY INFO
# Direct link to source data set ->
https://www.kaggle.com/datasets/
#adityajha1504/chesscom-user-games-60000-games
# Data set file: "club_games_data.csv" (166.67 MB) raw
# API command to download file -> kaggle datasets download -d
adityajha1504/
#chesscom-user-games-60000-games
# If error in fetching data set, kindly edit appropriate local
path in read.csv
```

```
#1. LOADING REQUISITES
require(magrittr)
require(tidyverse)
require(knitr)
require(lessR)
df = read.csv('/Users/guts/Documents/R Programs/MTHM501/Week
           5 Assessment/ chessdotcom_games_data.csv')
# Chess.com data set with 65k+ obs, 14 features.


#2. DATASET EXAMINATION: By Perceiving Categorical
Features gameVariantsList = unique(df$rules) # 5 unique
game variants resultsList = unique(df$white_result) # 13
unique game outcomes timeClassesList =
unique(df$time_class) # 4 unique time classes
timeControlsList = unique(df$time_control) # 62 unique
time controls


#3. DATA WRANGLING & TINKERING: To Prepare For Some Data
Science!
df = df %>% filter(rated=='True' & rules=='chess' & # filtered
to rated chess white_result != 'abandoned' &
   black_result != 'abandoned') %>% # excluded abandoned games
  dplyr::rename(white_elo = white_rating,
        black_elo = black_rating) %>% #
        renamed cols
  select(c(white_elo, black_elo, white_result,
         black_result, time_control)) %>% # selected
         required features
  mutate(white_elo = as.numeric(white_elo), black_elo =
        as.numeric(black_elo)) # mutated all elos to
        numeric


#4. MISSING VALUES: Considering 2 Essential Columns Below As
Missing Features result = c() # to minimize 2 complex result
```

```
features into 1 simplified feature white_outcome = c() # to
create numeric feature that quantifies result feature

for(i in 1:nrow(df)) { # Categorizing result data; given
white's perspective
  resultValue = switch(df$white_result[i],
                'win'            ='white-won',
               'checkmated'      ='black-won',
               'timeout'         ='black-won',
               'resigned'        ='black-won',
                'timevsinsufficient' ='draw',
               'repetition'       ='draw',
              'insufficient'     ='draw',
               'stalemate'       ='draw',
               'agreed'          ='draw',
               '50move'          ='draw')
                                    # draw = 0,
  outcomeValue = switch(df$white_result[i], # white-win = +1,
  black-win = -1
                'win'             = +1,
               'checkmated'       = -1,
               'timeout'          = -1,
               'resigned'         = -1,
                'timevsinsufficient' = 0,
               'repetition'       = 0,
              'insufficient'     = 0,
               'stalemate'       = 0,
               'agreed'          = 0,
               '50move'          = 0)
  result = append(result, resultValue) # inserting values
into new features white_outcome = append(white_outcome,
outcomeValue) }
df = df %>% mutate(match_rating = (df$white_elo +
df$black_elo)/2, #mean\median
                delta_elo = df$white_elo - df$black_elo,
                #difference in elo result = result, # added
                derived result feature to data set
```

```
                white_outcome = white_outcome) %>% # quantified
                result col
  select(-c(white_result, black_result)) # removed 2 cols, no
  longer needed

timeControlsDf = data.frame(table(df$time_control)) %>% # time
control freq df arrange(-Freq) # sorting by frequency to filter
                  a single best time control

timeControlTable = kable(timeControlsDf) # tabulated time
controls frequency timeControlTable # 2nd most popular time
control selected as 1st is a 1min game

df = filter(df, time_control=='600') %>% # Filters to 10 min
  time control select(-time_control) # Removing time_control as
  all values are 600 sec


#5. GRAPHICS: Scatterplot, LinearModel, Histograms, PieCharts
# plotting PvP elo from rated matchmaking
matchmakingScatterplot =
  ggplot(df, aes(x=black_elo, y=white_elo, col=delta_elo)) +
  ggtitle('Match-Making Elo Scatterplot') +
  geom_point(size=0.25) + labs(colour='Elo Difference') +
  xlab('Black Elo') + ylab('White Elo') +
  scale_colour_gradient2(low = 'black', mid = 'red',
  high='white') + theme_classic() + theme(panel.background =
  element_rect(fill='darksalmon'))

matchmakingLinearModel =
  ggplot(df, aes(x=black_elo, y=white_elo, col=result)) +
  geom_point(size=0.5) + ggtitle('Match-Making Result Linear
  Regression') + xlab('Black Elo') + ylab('White Elo') +
  theme_classic() + scale_color_manual(values = c('black',
  'red', 'azure')) + theme(panel.background =
  element_rect(fill = 'bisque')) + stat_smooth(method='lm',
  col='green') # linear regression on matchmaking
```

```
# histograms will help visualize games played across elo
matchRatingDensity = density(df$match_rating) # kernel density
estimate value plotGameVolumesHistogram = function(kde=FALSE,
d=matchRatingDensity) {
  hist(df$match_rating, breaks = 15, freq = !kde,
       main = 'Volume of Games Played Across
       Ratings', xlab = 'Match Ratings (Average of 2-
       player elo in versus)', ylab = 'Volume of
       Games Played',
       col='aquamarine')
  if(kde) {lines(matchRatingDensity)}
  axis(side = 1, at = 100)
}

matchmakingScatterplot # running scatterplot
matchmakingLinearModel # running linear regression on
scatterplot

plotGameVolumesHistogram()
plotGameVolumesHistogram(kde = TRUE,
d=matchRatingDensity) # histograms indicate peak
player volumes in 1200-1600 elo range

# filtering for maximum game volumes for players matched at
closer elo highGameVolumesDf = filter(df, match_rating>1200 &
match_rating<1600 &
                    delta_elo>=-10 & delta_elo<=10)
                                              #1065 games found

# plotting PieChart to visualize outcome distribution in high
volume games highGameVolResults = highGameVolumesDf$result #
extracted results plotOutcomePiechart_atHighGameVolumes =
function() {
  style(panel_fill = 'skyblue')
  PieChart(highGameVolResults, hole=0, density = 75,
         clockwise = TRUE, main='Outcome from Games
         Rated 1200-1600', fill =
```

```
        c('black','red','azure'), color = c('black'),
        values_color = c('white', 'white', 'black'))
} plotOutcomePiechart_atHighGameVolumes() # on

total 1065 games

#6. HYPOTHESIS TEST: Successfully Rejected Null Hypothesis
# Null Hypothesis -> "Actual outcome of a game can be
predicted by # (+\-)delta_elo/match_rating."

# (+\-)delta_elo is signed difference in elo of 2 players
matched together: # (white_elo - black_elo) & match_rating
is mean of both players' elos.

expectedOutcome = highGameVolumesDf$delta_elo/
                highGameVolumesDf$match_rating
                                # lies in a range of (-1 to 1)
actualOutcome = highGameVolumesDf$white_outcome # 3 discrete
values {-1,0,1}
# -1 is perceived as black-won
# 0 is perceived as draw
# +1 is perceived as white-won
predictiveOutcome_HypothesisTest = t.test(expectedOutcome,
                                  actualOutcome,
                                  alternative =
                                  'two.sided')
predictiveOutcome_HypothesisTest # runs test; outputs p.value
= 0.000644 # Since p.value observed << 0.05, null hypothesis
can be successfully rejected! # Alternate hypothesis is
supported by analysis of 1065 games, that outcome of # a game
CANNOT be predicted based on (difference in elo) / (match
rating).

# Therefore, each rating range can be analyzed in sub-groups
to
# conclude any possible advantage white can have in
higher rated games # --- E N D ---
```